LETTER

An Empirical Evaluation of Coverage Criteria for FBD Simulation Using Mutation Analysis

Dong-Ah LEE†, **Eui-Sub KIM**†, *Nonmembers*, *and* **Junbeom YOO**†a), *Member*

SUMMARY Two structural coverage criteria, toggle coverage and modified condition/decision coverage, for FBD (Function Block Diagram) simulation are proposed in the previous study. This paper empirically evaluates how effective the coverage criteria are to detect faults in an FBD program using the mutation analysis.
key words: *coverage adequacy criteria, FBD simulation, mutation analysis*

1. Introduction

Software testing is one of indispensable activities in the software development process. Software testing methods are traditionally divided into functional (black-box) testing and structural (white-box) testing, in which test cases are derived from program specifications and from the structure of programs, respectively. Functional testing verifies the functional correctness of software in any step of the software development process. Structural testing not only verifies the functional correctness but also measures coverage which means what percent of code has been exercised when a test suite runs. One or more structural coverage criteria, such as statement coverage, branch coverage, and condition coverage, are used to measure the coverage.

FBD (function block diagram) [1] is a commonly used programming language to develop software for PLC (programmable logic controller). Safety-critical systems often use the FBD to design software for digital I&Cs (instrumentation and control system). For example, the KNICS (Korea Nuclear Instrumentation and Control System) project implemented trip (shutdown) logics of a BP (bistable processor) for RPS (reactor protection systems). Testing FBD software often performs simulation-based testing for functional verification. The previous study [2] proposed two sets of structural coverage criteria for simulation scenarios of FBD, *toggle coverage* (*TC*) and *modified condition/decision coverage* (*MC/DC*), also known as simulation coverages. The simulation coverages are similar to structural coverage criteria of software testing, however they are for structural elements of an FBD program and the software testing is for source codes.

This paper empirically evaluate the effectiveness of the simulation coverages [2] using mutation analysis. Software requires rigorous quality when developing safety critical systems such as digital I&Cs in nuclear power plants (NPPs). Simulation verifies functional correctness of the software written in FBD. The simulation requires strong criteria to improve confidence in thoroughness. The mutation analysis is a fault-based software analysis technique to measure the adequacy of a test suite or the effectiveness of a adequacy criterion [3], [4]. The analysis seeds artificial faults (mutations) into an FBD program, then the simulation with a set of scenarios tries to detect the faults in the FBD program having one of the faults (mutants). If the scenarios achieving a higher percent of coverage finds more mutants than ones with a lower percent, the coverage criterion is effective to detect the faults.

The analysis uses trip (shutdown) logic programs [5] of BP, which is a part of the RPS developed in the KNICS project [6]. We generated three types of simulation scenarios, random, guided, and manual scenarios. Results of the mutation analysis show that simulation scenarios achieving a higher percent of both coverages (*TC* and *MC/DC*) detect more mutants than ones achieving a lower percent. In other words, the both coverage criteria are suitable for use as a measure of whether simulation scenarios are sufficient to detect faults in an FBD program.

The remaining part of the paper proceeds as follows: Sect. 2 briefly introduces the structural coverage criteria for FBD simulation and mutation analysis. Section 3 gives a full explanation of research questions and evaluation process and Sect. 4 explain analysis results of the evaluation. Finally, Sect. 5 concludes the paper and provides remarks on future research.

2. Background

2.1 Structural Coverage Criteria for FBD Simulation

FBD, one of the five standard PLC programming languages [1], is a commonly used graphical language to develop software for safety-critical systems. For example, the KNICS project used FBD to implement control software of NPPs [6]. The FBD program in Fig. 1 is a simplified trip (shutdown) logic. It has 5 blocks (2 `LT`, 2 `AND`, and 1 `OR` blocks), 7 inputs (3 integer `(I)` and 4 boolean `(B)` inputs), and 1 boolean output (`TRIP`). `RNG_MIN` and `RNG_MAX` are constants fixed with 10 and 20,000 respectively.

Simulation verifies that an FBD program is function-

Manuscript received June 19, 2020.
Manuscript revised September 4, 2020.
Manuscript publicized October 9, 2020.
†The authors are with the Division of Computer Science and Engineering, Konkuk University, Seoul, the Republic of Korea.
a) E-mail: jbyoo@konkuk.ac.kr
DOI: 10.1587/transinf.2020EDL8090

ally correct. Thoroughness, quality, or effectiveness of simulation scenarios are important to increase confidence of the simulation. Structural coverage criteria for FBD simulation [2], [7] improve and refine the simulation scenarios quantitatively.

(1) Toggle coverage (TC)

TC measures how many boolean outputs of blocks in an FBD program are changed from a value of zero to one (0-to-1) and back from one to zero (1-to-0) during simulation. An output is fully covered, *i.e.* 100% *TC*, when it toggles back and forth at least once. For example, the output of the block `(1) LT_INT` in Fig. 1 is fully covered, when the `PV_OUT` has 5, 15, and 5 sequentially. Simulation of an FBD program uses a set of simulation scenarios, and *(TC)* measures all toggles during the simulation with massive scenarios. For instance, the FBD program in Fig. 1 has 10 possible toggles which the five blocks have two possible toggles. If a set of simulation scenario toggles all possible ones, then simulation using the scenarios achieves 100% *TC*.

(2) Modified condition/decision coverage (*MC/DC*)

MC/DC measures how many important combinations of blocks in an FBD program simulation covers. The important combinations means sets of inputs for a condition of a block which independently affects an output of the block. For example, combinations of inputs of `(3)AND_BOOL` block has four possible input sets ((0,0), (0,1), (1,0), (1,1)), however only three combinations (*i.e.*, (0,1), (1,0), (1,1)) are important combinations. *MC/DC* counts all the important combinations executed with respect to a set of simulation scenarios along the same way as *TC*.

2.2 Mutation Analysis

Mutation analysis, which is often called mutation testing for software testing, is one of software analysis techniques to measure the adequacy of a test suite or the effectiveness of a test adequacy criterion. Research activities about techniques and tools of mutation analysis are increasing, and applicability is getting widespread [3], [8]. A mutant is a modified version of an original program, which has an artificial fault. The mutation analysis tries to detect the mutant using test suites—distinguish the behavior of the mutant from that of the original one—and evaluates the adequacy of the test suites. If a test suite is adequate for an test criterion and detects mutants as much as the adequacy, the test criterion is effective to assess quality of the test suite. Using mutation analysis, this paper analyzes coverage criteria for FBD simulation proposed in the previous work [2].

Shin [12] [13] analyzed FBD test coverage criteria using mutation analysis. The criteria measure coverages about testing of FBD programs. Each test case independently executes an FBD program, which means that the test cases do have single scan cycle. The criteria in this paper are for simulation-based testing of FBD programs, however. The simulation executes an FBD with the use of simulation scenarios which have multiple scan cycles. It is worth noting that it is necessary to execute FBD programs with multiple scan cycles to verify the function correctness because PLC programs are executed in a permanent loop.

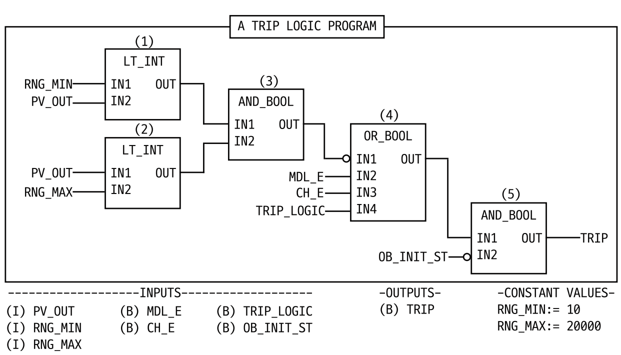

Fig. 1 A small FBD program for a trip (shutdown) logic

3. Empirical Design

3.1 Research Questions and Subjects

This paper investigates the following research questions:

- **RQ1**: How effective is *TC* for FBD simulation in fault detection?
- **RQ2**: How effective is *MC/DC* for FBD simulation in fault detection?

To answer the questions, we designed our experiments as described in Fig. 2. We generated three types of simulation scenarios, random (S_r), guided (S_g), and manual scenarios (S_m), for an original FBD program using an automated

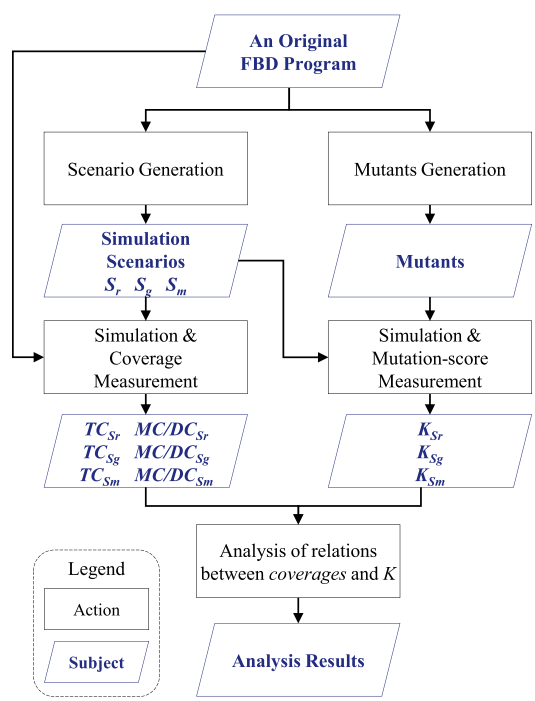

Fig. 2 The demonstration process and targets of research questions for RQ1

Table 1 Mutation operators for an FBD program

ID	Operator	Description	Constraint
CVR	constant value replacement	replace a constant value $C1$ by $C2$	$C1 \neq C2$
ABR	arithmetic block replacement	replace arithmetic block ψ with arithmetic block ϕ	$e_1 \psi e_2 \neq e_1 \phi e_2$
LBR	logical block replacement	replace logical block ψ with logical block ϕ	$e_1 \psi e_2 \neq e_1 \phi e_2$
CBR	comparison block replacement	replace comparison block ψ with comparison block ϕ	$e_1 \psi e_2 \neq e_1 \phi e_2$
IVR	input variable replacement	swap an input variable V_I with another	$\mid V_I \mid \geq 2$

Table 2 Summary of *TC*, *MC/DC*, and *K* for the three types of simulation scenarios

Name of FBD programs	Number of Mutants	Types of simulation scenarios								
		S_r			S_g			S_m		
		TC_{Sr}	MC/DC_{Sr}	K_{Sr}	TC_{Sg}	MC/DC_{Sg}	K_{Sg}	TC_{Sm}	MC/DC_{Sm}	K_{Sm}
FFT	43	11	40	55	64	74	83	85	91	90
VFT	63	15	42	55	66	72	84	88	91	90
MFT	67	15	45	55	66	72	83	88	92	91
FRT	43	12	40	55	65	73	83	87	91	90
VRT	63	14	42	55	66	72	84	87	92	90

tool, *FBDScenaGen+* [9]. The processing value, which is a name of processing data for the FBD program, has random values in the S_r. On the other hands, the value in the S_g has guided values, such as an increase or decrease. The S_m is manually generated by an domain expert. We simulated an original FBD program with each set of simulation scenarios and measured *TC* and *MC/DC* using *FBDCover* [7]. Meanwhile, we applied mutation operators in Sect. 3.2 to the original FBD program in order to generate a number of mutants. We also simulated the mutants, i.e., faulty FBD programs, with the three sets of simulation scenarios and measured how many mutants the scenarios detect. Finally we analyze them in order to answer the research questions.

The experiment uses FBD programs [5] for the second phase of KNICS APR-1400 RPS BP [6] as an original FBD program. It was excerpted from an almost (but, not officially final version) commercial NPP in operation. The BP consists of 18 shutdown logics written in FBD, but we only use 5 representative trip programs, 'fixed set-point falling trip' (FFT), 'variable set-point falling trip' (VFT), 'manual reset falling trip' (MFT), 'fixed set-point rising trip' (FRT), and 'variable set-point rising trip' (VRT), in this experiment.

3.2 Mutant Generation and Mutation-Score Measurement

Mutants should be plausible as faulty programs. In other words, the faults represent mistakes that programmers may make. The mutants are created by seeding such faults following a pattern which is called *mutation operators*. We defined mutation operators for an FBD program base on earlier research [10]. Table 1 lists five mutations operators. The list includes common mistakes during FBD programming. It does not include faults which tools can identify, however. For example, 'FBD Checker' [11] identifies the type mismatch or missing links.

The mutation analysis measures that how much mutants a set of simulation scenarios detect during a simulation, called a mutation-score. The mutation-score, *K*, is described as follows:

$$K = \frac{a\ number\ of\ detected\ mutants}{a\ number\ of\ total\ mutants} \times 100(\%)$$

We measure the *K* for each set of simulation scenarios for each program. If a set of simulation scenarios finds all mutants, then the *K* is 100%. Table 2 indicates the numbers of mutants for each FBD programs we generated.

4. Analysis Results

A simulation scenario has 100 execution cycles and a set (S_r, S_g, S_m) includes 1,000 simulation scenarios. Each of the sets reports *TC*, *MC/DC*, and *K* of the 5 FBD programs individually. S_r achieved 10–15% TC_{Sr} and 40–45% MC/DC_{Sr} while S_g achieved 64–66% TC_{Sg} and 72–74% MC/DC_{Sg}. The results of S_r and S_g means that a set of simulation scenarios which is generated by a guidance is more effective to achieve the both coverages than one randomly generated. S_m achieved 85–88% TC_{Sm} and 91–2% MC/DC_{Sm} against all the original FBD programs.

Mutant generation uses only one mutation operator in Table 1 to the original programs in order to generate one mutant. Tens of mutants are generated by the operators for every five programs. Table 2 shows that the number of generated mutants. We performed mutation analysis to the mutants with S_r, S_g, and S_m. S_r found about 55% of mutants, S_g found about 84% of mutants, and S_m found over 90% of mutants.

(1) RQ1: How effective is *TC* for FBD simulation in fault detection?

The graph (a) in Fig. 3 shows the correlation of *K* with *TC*. The vertical axis represents *K* and the horizontal axis represents *TC*. The line is given as relationship between the two variables. These two variables have a positive association because as the *TC* increases, so does the *K*. The value of Pearson's r is 0.9934 and they have linear relationship of

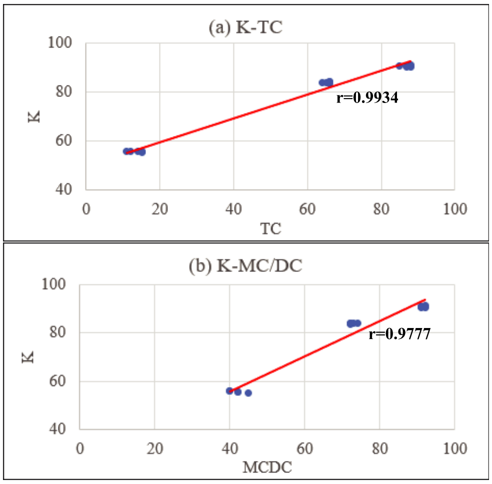

Fig. 3 The correlation of the mutation-score (K) with the coverage criteria (*TC* and *MC/DC*)

strong strength. In other words, the higher percent of *TC* a set of simulation scenarios achieves, the more mutants the set detects.

(2) RQ2: How effective is *MC/DC* for FBD simulation in fault detection?

The graph (b) in Fig. 3 shows the correlation of K with *MC/DC*. The value of r in this case indicates (r = 0.9777) that there is a positive and linear relationship of strong strength between the two variables. The higher percent of *MC/DC*, in common with *TC*, a set of simulation scenario achieves, the more faulty program the set finds.

The results give us information that both coverage criteria, *TC* and *MC/DC*, are effective to detect a fault in an FBD program. A set of simulation scenario usually achieves the higher percent of *MC/DC* than *TC*. Even if a set achieves a low percent of about 10% *TC*, the set finds more than half of mutants. In order to detect faults sufficiently, however, the set should achieve a sufficiently high percent of both criteria. *TC* should be over about 87% and *MC/DC* should be over about 91% to detect over 90% of faults in the experiment.

One of the results indicates that the detection of mutants by CBR is relatively more difficult than others. The mutation-score S_r for LBR is about 66% and S_r for CBR is about 25% for FFT. S_g for LBR is about 89% and S_g for CBR is about 63%. The reason is that some of mutants by CBR make a slight difference, such as generating `LE_INT` from `LT_INT`. This means that it is difficult to detect these kinds of fault by simulation-based testing.

5. Conclusion and Future Work

This paper reports empirical evaluations for FBD simulation coverage criteria by mutation analysis. We used 5 representative FBD programs for the evaluation and generated tens of mutants for each programs using mutation operators. The experimental results demonstrated that simulation scenarios which achieve a higher percent of coverages are more effective to detect faults in an FBD program.

The most important limitation lies in the fact that achieving a sufficiently high percent of coverages takes lots of time and effort. It was possible to generate simulation scenarios achieving over 87% *TC* and 90% *MC/DC* by domain experts manually. Although *FBDScenaGen+* generated a number of simulation scenarios automatically, they only detected under 84% of mutants. We have a plan to improve the scenario generation using a machine learning technique in our future work.

Acknowledgments

This research partially supported by Next-Generation Information Computing Development Program through the National Research Foundation (NRF) of Korea funded by the Ministry of Science, ICT (NRF-2017M3C4A7066479).

References

[1] "IEC 61131-3: Programmable controllers—Part 3: Programming languages," International Electrotechnical Commission (IEC), 2013.
[2] D.-A. Lee, E.-S. Kim, and J. Yoo, "Quantitative Measures of Thoroughness of FBD Simulations for PLC-based Digital I&C System," Nucl. Eng. and Technol. (in press). DOI:10.1016/j.net.2020.06.017
[3] Y. Jia and M. Harman, "An Analysis and Survey of the Development of Mutation Testing," IEEE Trans. Softw. Eng., vol.37, no.5, pp.649–678, Sept. 2011. DOI:10.1109/TSE.2010.62
[4] J.H. Andrews, L.C. Briand, Y. Labiche, and A.S. Namin, "Using Mutation Analysis for Assessing and Comparing Testing Coverage Criteria," IEEE Trans. Softw. Eng., vol.32, no.8, pp.608–624, Aug. 2006. DOI:10.1109/TSE.2006.83
[5] Korea Atomic Energy Research Institute (KAERI), "Software design specification for reactor protection system KNICS-RPS-SD231," Rev.02, 2006.
[6] KAERI, "Software requirements specification for reactor protection system KNICS-RPS-SRS221," Rev.00, 2006.
[7] E.-S. Kim, S. Jung, J. Kim, and J. Yoo, "MC/DC and toggle coverage measurement tool for FBD program simulation," Trans. of the Korean Nucl. Society Spring Meeting, May 2016.
[8] M. Papadakis, M. Kintis, J. Zhang, Y. Jia, Y.L. Traon, and M. Harman, "Mutation testing advances: an analysis and survey," Adv. in Comput., vol.112, pp.275–378, Elsevier, 2019. DOI: doi.org/10.1016/bs.adcom.2018.03.015
[9] E.-S. Kim, S. Jung, J. Yoo, Y.J. Lee, and J. Lee, "FBDScenaGen+: GA-based high-quality scenario generator for FBD simulation," Int. Symp. on Future I&C for Nucl. Power Plants, Gyungju, the Republic of Korea, 2017.
[10] Y. Oh, J. Yoo, S. Cha, and H.S. Son, "Software safety analysis of function block diagrams using fault trees," Reliab. Eng. & Syst. Safe., vol.88, no.3, pp.215–228, June 2005. DOI:10.1016/j.ress.2004.07.019
[11] S. Jung, J. Yoo, and Y. Lee, "A PLC platform-independent structural analysis on FBD programs for digital reactor protection systems," Ann. Nucl. Energy, vol.103, pp.454–469, May 2017. DOI:10.1016/j.anucene.2017.02.006
[12] D. Shin, E. Jee, and D.-H. Bae, "Empirical evaluation on FBD model-based test coverage criteria using mutation analysis," Int. Conf. on Model Driven Eng. Lang. and Syst., Springer, pp.465–479, 2012.
[13] D. Shin, E. Jee, and D.-H. Bae, "Comprehensive analysis of FBD test coverage criteria using mutants," Softw. Syst. Model 15, pp.631–645, 2016. DOI:10.1007/s10270-014-0428-y